HOW TO TRAIN YOUR WORLD MODEL: FINE-TUNING VS RAG FOR LM-BASED WORLD MODELING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models (WMs) simulate the transition dynamics of environments, enabling agents to plan over the consequences of their actions. In text-based environments, fine-tuning a Language Model (LM) to serve as a WM has emerged as a dominant paradigm. However, despite the widespread success of non-parametric approaches such as Retrieval Augmented Generation (RAG), retrieval over past experience for LM-based world modelling remains underexplored. We conduct a systematic evaluation across five diverse environments spanning embodied, web navigation and social settings, comparing fine-tuning and RAG-based approaches for LM-based world modelling. Our study reveals that fine-tuning often outperforms RAG, with fine-tuned WMs enabling agents to obtain higher rewards on 15/20 settings. However, there are several cases where RAG proves superior, and there are often cases where simple prompting baselines match or outperform finetuning. While both construction paradigms benefit from additional and more diverse exploration, RAG-based approaches prove more data-efficient, while fine-tuning approaches disproportionately benefit from scaling the amount of experience collected. With a focus on RAG-based WMs, we devise a procedure that uses counterfactual intervention to estimate the error rate of the retrieval stage, and show that retrievers consistently surface suboptimal transitions from the experience buffer. Hoping to address this failing, we study a variety of query reformulation strategies, demonstrating that a hierarchical approach outperforms the traditional retrieval pipeline. Finally, we compose our findings into a hybrid world modelling system that parametrically captures core environment dynamics, while learning to rely on retrieval from an actively maintained memory store. Our hybrid system consistently outperforms other methods across multiple environments and models, showcasing the robustness of the approach and the applicability of our findings [1].

1 INTRODUCTION

Agentic systems increasingly improve by scaling experience within complex environments, shifting away from a reliance on dense human annotation and engineered reward systems (Silver & Sutton, 2025). A dominant approach to scaling experience is the development of *world models* (WMs)—internal simulators that learn the underlying transition dynamics of an environment (Ha & Schmidhuber, 2018). Using a WM, agents can go beyond simply reacting to the current state and instead plan by considering the consequences of their actions before executing them.

To leverage the rich knowledge priors encoded in Language Models (LMs) (Petroni et al., 2019), recent work has explored using LMs as the backbone for world models (Gu et al., 2024). These methods follow the traditional approach to world modelling of collecting exploratory trajectories on a training set, and then finetuning a model to predict future observations (Watter et al., 2015; Chae et al., 2025). When applied to LMs, this parametric approach yields WMs that can boost agent performance on a variety of environments, from embodied settings to web-domains (Li et al., 2026).

Particularly prominent in the LM domain is the use of non-parametric approaches, such as retrieval-augmented generation (RAG) (Arslan et al., 2024). Relying on the in-context capabilities of LMs, RAG make no changes to the weights, but rather provides examples and additional information relevant to the task at hand to help ground the LM's response. While both fine-tuning and

[1] Code will be made available after the review period

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

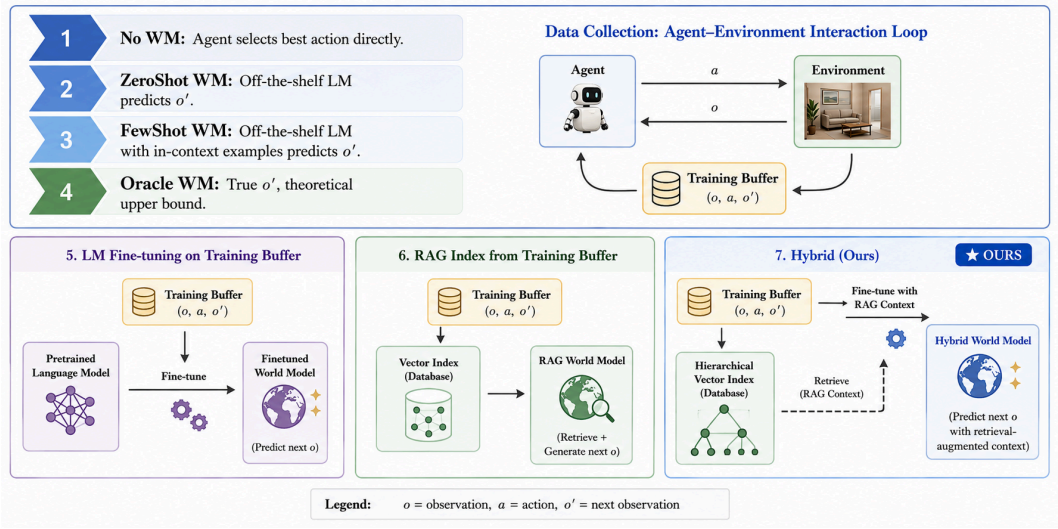

Figure 1: We investigate the performance of fine-tuning vs RAG for the task of LM-based world modeling. Finally, we distill our insights into a hybrid methodology that combines fine-tuning with a hierarchical index design to robustly improve the performance of LM-based WMs.

RAG have their individual strengths (Balaguer et al., 2024), direct comparisons demonstrate that RAG-based approaches can often substantially outperform parametric adaptation in knowledge-acquisition tasks (Ovadia et al., 2024; Soudani et al., 2024). However, despite the power of non-parametric approaches, their application to LM-based world modelling remains underexplored.

In this work, we systematically compare strategies for constructing LM-based WMs across five environments, including embodied tasks (ALFWorld (Shridhar et al., 2020), TextWorld (Côté et al., 2018)), web navigation (WebShop (Yao et al., 2022)), and social conversations (Sotopia (Zhou et al., 2024), Chameleon (Karabag et al., 2025)). We construct both fine-tuned and RAG-based WMs and compare their performance on the axes of WM fidelity, as well as agent utility (Section 3.2).

Prior studies comparing RAG and fine-tuning frequently find RAG to be dominant in tasks such as knowledge-grounded question answering (Ovadia et al., 2024). Our results show the opposite: RAG underperforms fine-tuning on 15/20 model-environment combinations, but maintains a strong lead on specific environments. Additionally, on 9/20 settings, either fine-tuning or RAG is outperformed by a simple prompting baseline, demonstrating that the ideal approach is highly dependent on the specific environment in question.

Ablations over the buffers used to construct the WMs reveal approach-specific trends. Both RAG and fine-tuning benefit from larger and more diverse exploration data; however, RAG is more data-efficient, while fine-tuning exhibits more persistent returns to increases in scale.

With a focus on the yet unexplored RAG-based WM approach, we devise a counterfactual intervention experiment to estimate the error rate of the retrieval stage. Our findings are stark and clear: the retrieval step fails consistently with RAG-based WMs, with recall@5 falling below 0.7. Motivated by this finding, we investigate more complex ways of executing retrieval over observations and actions. We find that a hierarchical retrieval paradigm, which selects based on observation similarity before re-ranking based on action similarity, outperforms all other approaches.

Finally, we introduce a hybrid paradigm which combines the parametric and RAG-based approaches to construct a world model. Our hybrid approach outperforms every other method on 17/20 model-environment combinations, approaching 77% of the utility of an oracle upper bound (compared to 60% by the next best method). On 45% of environment-model combinations, our hybrid exceeds 90% of the performance of the oracle model, establishing the power of the method.

2 Background and Related Work

We tackle the generic problem of an agent in an environment, formally represented as a partially observable Markov decision process (POMDP) (Puterman, 1990), defined by the 7-tuple:

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O}, \gamma \rangle \tag{1}$$

where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and Ω is the observation space. The transition probability function $\mathcal{P}(s' \mid s, a)$ denotes the probability of transitioning to state $s' \in \mathcal{S}$ given the current state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$. At each time step, the agent receives an observation $o \in \Omega$ determined by the observation function $\mathcal{O}(o \mid s', a)$, alongside a scalar reward from the reward function $\mathcal{R}(s, a)$. Finally, $\gamma \in [0, 1)$ represents the discount factor. Given a POMDP, the goal of an agent is to select actions at each step such that the expected total reward is maximized:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t) \right] \tag{2}$$

where $\tau = (o_0, a_0, \ldots a_{t-1}, o_t)$ denotes a trajectory sampled from the environment under policy π.

2.1 World Models

World Models estimate the transition function $\mathcal{P}(s' \mid s, a)$ (Ha & Schmidhuber, 2018). However, since agents are only exposed to observations, world models typically learn a transition function based on observations, $\mathcal{P}(o' \mid o, a)$. World models have a variety of uses, including the replacement of expensive environment calls during agent learning (Hafner et al., 2020), and the enabling of planning over the potential consequences of an agent's selected actions (Sekar et al., 2020).

The traditional approach to training a world model has its roots in image generation (Hafner et al., 2019). First, in the exploration phase, agents collect an experience replay buffer on training splits of the environment $\mathcal{D} = \left\{ \tau^{(i)} \right\}_{i=1}^{N}$. We then use the transitions from the buffer to train a world model: $\text{WM} : \mathcal{O} \times \mathcal{A} \to \mathcal{O}$ by minimizing the prediction error: $\mathcal{L}(\theta) = \mathbb{E}_{\tau \sim \mathcal{D}} \left[\text{loss}(\text{WM}_\theta(o_t, a_t), o_{t+1}) \right]$.

Since their emergence as powerful aggregators of human knowledge and reasoning skills, there has been increased interest in using LMs to serve as the foundations for WMs (Gu et al., 2024; Chae et al., 2025). These works follow the traditional paradigm, and fine-tune a LM to predict the next observation with a next-token-prediction loss.

A strong alternative to finetuning is non-parametric adaptation or retrieval-augmented generation (RAG), often used to facilitate memory in agentic systems (Kim & Hwang, 2025; Ferraz et al., 2026). Work on directly using RAG for world modeling has been limited to initial explorations. Prior work (Mei et al., 2025) creates WMs with RAG mechanisms over external, tutorial data to directly reduce the degradation of long-horizon reasoning. However, this paradigm requires access to manually curated external datasets, a resource which is hard to gather for many complex environments. To the best of our knowledge, the only work that uses RAG over self collected experience replay buffers to construct LM-based WMs are Yang et al. (2025) and Zhang et al. (2026), however they do not compare against fine-tuning, leaving the comparative strength of the approach unproven.

2.2 Systematic Comparisons of LM-based World Models

Prior work in domains such as knowledge acquisition has often directly compared fine-tuning and RAG-based approaches. While neither method is strictly superior to the other (Balaguer et al., 2024), there is widespread agreement that RAG-based approaches enable more effective knowledge acquisition, especially for less popular or domain-specific knowledge (Ovadia et al., 2024; Soudani et al., 2024; Pingua et al., 2025). Existing comparisons of LM-based WMs such as Li et al. (2026), establish a comprehensive evaluation framework for fine-tuned LM-based world models, demonstrating that fine-tuning substantially enhances a LM's next-state prediction fidelity and long-horizon consistency. However, none of these works consider any RAG-based WMs in their comparisons, leaving open the question of how fine-tuning performs relative to RAG in this domain. Our work addresses this gap and conducts the first systematic comparison of fine-tuning and RAG-based approaches for the task of LM-based world modeling.

3 FINE-TUNING V.S. RAG FOR WORLD MODELING

To facilitate a comprehensive evaluation, we compile a diverse testbed of five text-based environments spanning the domains of embodied navigation, web navigation, and social conversations.

Embodied Navigation: These environments benchmark embodied decision-making within text-simulated physical spaces. Specifically, we use the ALFWorld and TextWorld environments. In ALFWorld, agents complete household tasks (e.g., "put a clean apple in the fridge") by navigating rooms and interacting with objects. TextWorld similarly offers procedurally generated text-based quests, such as gathering ingredients to cook a specific recipe. Agents receive a sparse binary reward $(+1)$ upon the successful completion of the high-level task. In these tasks, the WM must learn to predict updated room descriptions, inventory state, or interaction outcomes.

Web Navigation: We use the WebShop environment, a simulated e-commerce environment where agents must navigate a web interface to purchase specific items based on natural language instructions (e.g., "buy a red men's running shoe size 10 under $50"). The agent interacts by issuing search queries and clicking text-based buttons. The agent receives a dense scalar reward between 0 and 1 based on the attribute and semantic overlap between the purchased item and the target instruction. The WM must learn to predict search results pages or product details pages.

Adversarial Negotiation: To evaluate the agent's ability to achieve complex social objectives, we simulate two-agent, mixed-motive conversations using scenarios adapted from SOTOPIA. The agent must negotiate effectively against a simulated partner (powered by an LM) endowed with hidden information and competing priorities. For example, the agent may be tasked with securing a $40 loan from a friend who is secretly saving for a major purchase. The agent receives a scalar reward proportional to the resource concessions obtained from the conversation partner by the end of the dialogue. Given the conversation transcript history and the agent's proposed next utterance, the WM aims to predict the simulated partner's subsequent conversational response.

Deceptive Communication: This environment tests the agent's capacity for social reasoning and managing implicit information flow via the social deduction game "Chameleon." In this game, players receive a secret word, except for one player (the Chameleon). Players take turns stating a single related word to prove they know the secret without revealing it to the Chameleon. We simulate a 3-player game where the agent always acts first and is a standard player (non-Chameleon). This reduces to a single-turn decision problem: selecting a word that balances signalling knowledge to peers while obfuscating it from the Chameleon. After every player has stated a word, an external judge guesses which one was the Chameleon. The agent is rewarded if the Chameleon is correctly accused by the judge, is given zero reward if the other (non-Chameleon) player is accused, and a negative reward if it is accused. Since this is a single-step environment, the WM is trained to predict the subsequent words generated by the other two players, as well as the judge's verdict, conditioned on the agent's chosen word.

For implementation details, refer to Appendix A.

3.1 METHODS

We evaluate a range of baseline LM world models. These methods do not require active construction and serve as reference points for the performance of fine-tuned or RAG-based WMs:

No WM (No): This setting acts as our foundational, model-free baseline where the agent operates entirely without a world model. Instead of rolling out simulated future states, the agent's policy directly selects an action based purely on the current observation o_t.

Zero-Shot WM (ZS): We evaluate the inherent, out-of-the-box reasoning capabilities of pre-trained LMs. In this setting, the LM is prompted with the current observation o_t and action a_t, and is tasked with predicting the subsequent state o_{t+1} relying solely on its pre-trained parametric knowledge.

Few-Shot WM (FS): To account for the in-context learning capabilities of modern LMs (Brown, 2020), we augment the prompt with $k = 5$ randomly sampled environmental transitions (o_i, a_i, o_{i+1}). This evaluates whether standard few-shot demonstration is sufficient to accurately model environment dynamics without the need for targeted retrieval mechanisms.

Ground-Truth WM (Oracle): We query the actual underlying environment to perfectly resolve the next state. While unachievable in practical deployment, this oracle serves as a strict theoretical upper bound, isolating the downstream agent's performance given a flawless transition model.

The other methods require an active exploration phase where the agent collects an experience replay buffer on the training split of the environment $\mathcal{D} = \left\{\tau^{(i)}\right\}_{i=1}^{N}$. Given this buffer, we construct:

Fine-tuned WM (FT): We fine-tune the LM on verbalised transitions from the experience replay buffer. Specifically, we optimise the loss:

$$\mathcal{L}_{\text{NTP}}(\theta) = \mathbb{E}_{\tau\sim\mathcal{D}}\left[-\sum_{k=1}^{K}\log p_\theta\left(z_{t+1}^{k} \mid c_t, z_{t+1}^{<k}\right)\right] \tag{3}$$

where $c_t = \text{tokenize}(o_t, a_t)$ represents the serialised token sequence of the observation and current action, and $z_{t+1} = (z_{t+1}^1, \ldots, z_{t+1}^K) = \text{tokenise}(o_{t+1})$ denotes the sequence of K discrete tokens comprising the target next observation. The LM models the probability p_θ of each subsequent token z_{t+1}^k conditioned on the context prefix c_t and all preceding observation tokens $z_{t+1}^{<k}$.

Retrieval Augmented Generation WM (RAG): We construct an index over the transitions in the replay buffer by representing each transition as a key string: $o \circ a$, where $\circ$ is string concatenation. Each key string is embedded in a vector space with Nemotron-Embed-8B (Xu et al., 2025). During inference, given an observation o_t and candidate action a_t, we compute the embedding of $o_t \circ a_t$, retrieve the $k = 5$ most similar transitions $\{(o_i, a_i, o_{i+1}), \ldots\}$ from the experience replay buffer and serve them to the LM in-context as we request a prediction on the next observation.

3.2 World Model Deployment and Evaluation

We choose a consistent pipeline of deployment for all WMs. All of our experiments involve a frozen, off-the-shelf LM as a policy model/agent (in our study we use Llama3-8B (Grattafiori et al., 2024), Mistral-7B (Jiang et al., 2023), Gemma-4-26B-A4B (Team et al., 2025) and Gemma-4-31B) which uses a world model to plan over its next action. Specifically, the agent sees observation o_t and instead of predicting a single action a_t, predicts a list of candidate actions $a_{t,1}, \ldots a_{t,m}$. The world model under evaluation is used to predict the next observation for each of these candidate actions: $\tilde{o}_1, \ldots \tilde{o}_m$. We then use the policy-agent LM to estimate the value of the predicted observation on a scale of 1-10. The action which led to the most highly rated next observation is selected. In practice, we set $m = 5$. For exact prompts, hyperparameters and implementation details, see Appendix B. We follow prior work (Li et al., 2026) and evaluate each WM on two axes:

Fidelity: World models predict the next observation. A high-fidelity world model is one that more accurately predicts the next observation. Given that observations can be represented as strings, we use multiple measures of string similarity. Our metrics include lexical overlap (BLEU), semantic overlap (BertScore) and general meaning (Gemma-4-31B-as-a-judge on a scale of 1-10). However, initial results suggested that both BLEU and BertScore were problematic metrics, exhibiting little variance over runs with drastically different performance/behavior. Hence, we only report LM-as-a-judge in the main text. See Appendix D for complete results.

Utility: The eventual goal of world modelling is to create agentic systems which achieve higher reward. Hence, we additionally evaluate WMs by the total reward that an agent obtains when using it for planning. A WM with higher utility, by definition, leads to a higher boost in agent performance. To enable comparisons across environments, we scale all total rewards by the total reward obtained by an agent that has access to the oracle world model.

We compute 95% confidence intervals using a paired bootstrapped t-test (Koehn, 2004).

3.3 Results

When evaluating end-to-end task success, fine-tuning is the superior standalone method, achieving superior performance on 15/20 settings. RAG has its strengths; it maintains a consistent advantage over the baseline approaches and outperforms fine-tuning in the Chameleon environment.

Environment	Model	No	ZS		FS		FT		RAG	
		Util	Util	Fid	Util	Fid	Util	Fid	Util	Fid
ALF	Llama	0.36	0.43	4.15	0.25	3.96	**0.46**	**4.78**	0.36	4.09
	Mistral	0.00	0.30	3.95	0.20	3.82	**0.54**	**4.65**	0.39	4.12
	Gem-26B	0.79	0.80	4.33	0.76	4.16	**0.86**	**5.10**	0.81	4.87
	Gem-31B	0.82	0.67	4.23	0.72	5.01	**0.89**	**6.18**	0.86	5.11
Text	Llama	0.00	**0.16**	3.40	0.06	2.87	0.15	4.04	0.11	**4.15**
	Mistral	0.49	**0.63**	3.45	0.50	2.90	0.57	4.10	0.55	**4.20**
	Gem-26B	0.00	**0.22**	3.61	0.14	3.13	0.19	4.28	0.17	**4.36**
	Gem-31B	0.85	0.76	5.20	0.63	4.82	**0.92**	**5.86**	0.90	5.63
Web	Llama	0.04	0.07	4.77	0.09	3.78	**0.49**	**6.06**	0.24	5.08
	Mistral	0.00	0.17	4.65	0.25	3.85	**0.58**	**6.15**	0.33	5.15
	Gem-26B	0.08	0.12	4.81	0.17	4.11	**0.53**	**6.27**	0.35	5.34
	Gem-31B	0.90	0.60	4.01	0.69	4.29	**0.94**	**6.43**	0.92	5.81
Cha	Llama	0.17	−0.50	2.56	−0.10	2.70	−0.52	3.54	**0.33**	**3.87**
	Mistral	−4.50	−3.74	2.45	−3.07	2.60	−3.88	3.45	**0.64**	**3.95**
	Gem-26B	0.50	0.07	3.22	0.48	3.42	0.48	3.70	**0.66**	**4.31**
	Gem-31B	0.78	−0.05	3.28	0.41	4.16	0.73	3.83	**0.87**	**5.17**
Soto	Llama	0.75	0.89	4.50	0.82	4.19	**0.90**	4.61	0.86	**4.77**
	Mistral	0.95	0.96	4.40	0.96	4.24	**0.97**	4.58	**0.97**	**4.86**
	Gem-26B	0.85	0.87	4.61	0.87	4.42	**0.90**	4.92	0.89	**5.12**
	Gem-31B	0.96	0.96	4.73	0.64	4.62	**0.98**	4.81	0.97	**4.87**

Table 1: Utility and Fidelity of LM-based WMs, with best in **bold** and second best underlined. Fine-tuning is the superior standalone method, achieving superior utility on 15/20 settings. RAG maintains a consistent advantage over the baseline approaches and outperforms fine-tuning on the Chameleon environment. Results are significant at $p < 0.05$, see Appendix D for full table.

However, despite the superiority of actively constructed world models, they are not universally optimal. In 9/20 evaluated settings, either FT or RAG is outperformed by non-constructed baselines (No WM, ZS, or FS), reinforcing that the ideal approach remains highly environment-dependent.

Interestingly, when isolating the textual accuracy of the predicted next observations, the trend becomes less clear. While FT (4.87) slightly outpaces RAG (4.74) in raw generative accuracy, we notice that in environments such as TextWorld and Sotopia, RAG achieves greater average Fidelity than fine-tuned WMs, despite providing less Utility. This discrepancy correlates with findings from other studies on WMs (Li et al., 2026), which suggest that fidelity, while loosely correlated with total reward, can imply a different ranking over methods.

4 Ablations Over Experience Replay Collection

Since both fine-tuning and RAG-based WMs use replay buffers in their construction, we ask how the nature of the buffer affects the performance of each method. We vary the buffer over two axes:

Scale: We subsample the number of points in the buffer to evaluate how performance varies with data scale. Specifically, we construct WMs with experience replay buffers of n transitions with $n \in \{10, 20, 40, 80, 160, 320, 640, 1280, 2560\}$.

Exploration: Previous experiments constructed the buffers using an agent that greedily attempted to solve a task in the training environment. An alternate strategy is to mix exploitation and exploration and have the agent take unguided, exploratory actions in the environment $\epsilon\%$ of the time. This approach can potentially increase the diversity of states encountered during exploration, enhancing the WM's effective coverage. In practice, we implement the exploratory action selection mechanism with specialised prompts (see Appendix C) and set $\epsilon \in \{0, 25, 50, 75, 100\}$.

4.1 RESULTS:

We observe distinct scaling behaviours and exploration preferences between fine-tuning (FT) and retrieval-augmented generation (RAG) when varying the experience replay buffer.

Buffer Scale: Varying the size of the replay buffer reveals contrasting scaling trends. Across TextWorld and Sotopia, RAG achieves strong, near-peak performance with as few as $n = 40$ transitions. However, it tends to plateau or slightly degrade at higher scales. Fine-tuning fails at low scales, yielding near-zero rewards. It consistently reaches an inflexion point around $n = 80$, where performance rapidly accelerates and begins scaling predictably with more data.

Exploration Ratio: Injecting exploratory actions during buffer collection improves downstream world model performance, but the optimal distribution diverges by method. For both FT and RAG, completely unguided exploration ($\epsilon = 100\%$) and pure exploitation ($\epsilon = 0\%$) yield the lowest total rewards across all four environments. Fine-tuning universally peaks at $\epsilon = 75\%$ (e.g., reaching a total reward of 15.04 in TextWorld and 0.88 in Sotopia). The parametric model relies on encountering a vast, diverse set of edge-case states during training to accurately internalise the environment's transition dynamics. Conversely, RAG universally peaks at a moderate $\epsilon = 50\%$. We hypothesise that because RAG relies on similarity matching in a vector space, excessive random exploration introduces disconnected, noisy transitions that dilute the relevance of the retrieved context.

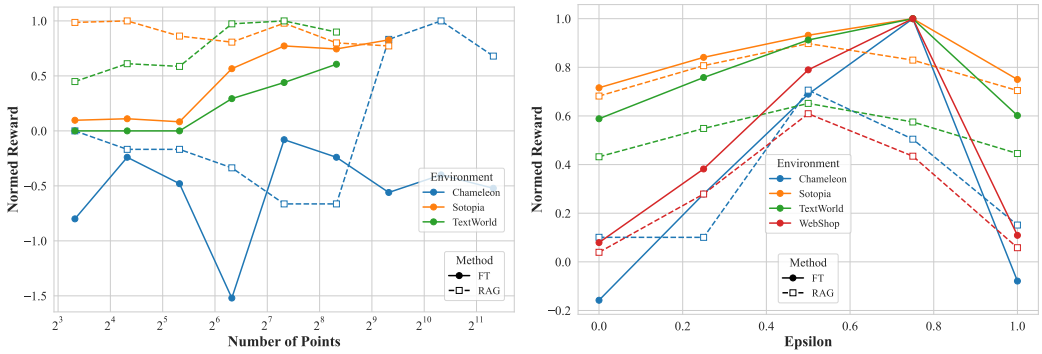

Figure 2: Data scale and exploration ϵ ablations. RAG is more data-efficient than FT, and both methods benefit from agent exploration.

5 DEEPER INVESTIGATION OF RAG-BASED WMS

Retrieval-augmented generation (RAG) introduces two distinct points of failure: the retrieval mechanism and the generation process. In standard information retrieval, documents can be evaluated against gold-standard annotations. However, transitions in an experience replay buffer lack pre-annotated relevance labels. Furthermore, a perfectly matching transition may not exist for a given observation-action pair, rendering the quantification of retriever error rates fundamentally ill-posed.

To isolate and evaluate the retrieval step, we introduce a pseudo-relevance scoring mechanism. Assume a world model $\text{WM}(o_t, a_t, e)$ that predicts the next observation given o_t, a_t, and a retrieved in-context demonstration $e = (o_i, a_i, o_{i+1})$. Let $\text{diff}(\hat{o}, o_{t+1})$ be an evaluation function (e.g., an LLM-as-a-judge) that quantifies the divergence between the predicted observation $\hat{o}$ and the ground-truth observation o_{t+1}. By holding o_t and a_t constant, we counterfactually evaluate candidate transitions from the replay buffer. A transition e_a is considered strictly better than e_b if:

$$\text{diff}(\text{WM}(o_t, a_t, e_a), o_{t+1}) < \text{diff}(\text{WM}(o_t, a_t, e_b), o_{t+1}) \tag{4}$$

By applying this function, we can empirically assign a pseudo-relevance score to every transition in the buffer, identifying the oracle transition that explicitly minimises the downstream prediction error. In practice, for each test instance, (o_t, a_t), we compute pseudo-relevance scores only for the top $k = 20$ most similar transitions $(e_1 \dots e_k)$ ranked by the cosine similarity between the text embedding of $o_t \circ a_t$ and $o_i \circ a_i, \forall i \in [1, k]$. Our metric of interest is Recall@k ($k \leq 5$), however we note that since we do not compute pseudo-relevance labels over the entire retrieval corpus, the

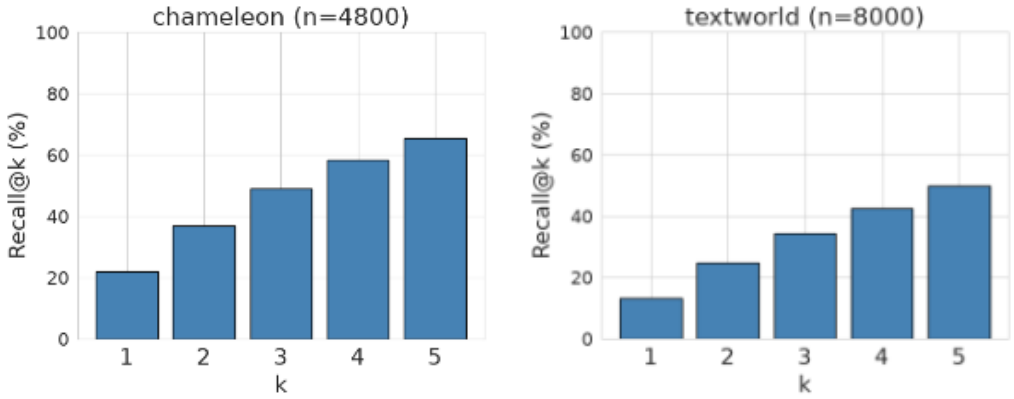

Figure 3: Estimated recall of the retriever on Chameleon and TextWorld. Recall@k is lower than 70% in both cases, suggesting a substantial gap in performance.

measures reported below are **upper-bounds** on the recall, i.e., if there is a superior trajectory outside the top-20, it would not affect the recall measure.

Evaluating dense retrievers against this oracle reveals severely degraded performance (Figure 3), with upper bounds on Recall@5 plateauing at merely 70%. This suboptimal recall indicates that off-the-shelf text embedders, trained primarily for semantic similarity, struggle to capture the dynamics of environment transitions. Consequently, there is headroom for optimisation in the retrieval phase.

To address these limitations, we investigate alternative query formulations. The naive approach concatenates the observation and action into a single query string. However, this forces semantically distinct modalities into a single representational space, frequently causing one feature to overshadow the other. We propose disentangling the representations through two alternative paradigms:

Linear Combination: We independently embed the observation and the action. The final retrieval score is a linear combination of their respective cosine similarities. While this successfully preserves individual modalities, its primary limitation is the reliance on a weighting hyperparameter, which is difficult to tune without a representative validation set. In practice, we weight components equally.

Hierarchical Retrieval: We structure retrieval as a two-stage cascade. First, we retrieve a broad candidate pool (e.g., top-50) based strictly on observation similarity. Second, we re-rank these candidates based on action similarity. This architecture reflects the intuition that the current state is the primary discriminant for environmental context, while the action serves as a secondary filter within similar states, avoiding the retrieval of identical actions in vastly different environments.

Results: Evaluating these retrieval paradigms across environments demonstrates that disentangling observation and action representations is generally beneficial, though the method of aggregation is critical. While a linear combination over independent embedding scores improves performance in WebShop (where it serves as the optimal method) and Chameleon, it actively degrades performance below the simple baseline in TextWorld (4.12) and Sotopia (0.514). Hierarchical retrieval proves more robust. The two-stage cascading approach yields the highest total reward in 3 out of 4 environments (TextWorld, Chameleon, and Sotopia). The improvement is particularly stark in TextWorld, where it nearly doubles the reward of the naive baseline (12.62 vs. 6.50). This confirms that prioritising observation similarity before filtering by action is an effective, parameter-free heuristic.

6 Hybrid World Model

While retrieval provides dynamic context and fine-tuning embeds domain-specific mechanics, both approaches exhibit isolated limitations. To synergise their strengths, we propose a Hybrid World Model (HWM) that integrates supervised fine-tuning with non-parametric retrieval, explicitly training the language model to condition its predictions on retrieved historical experiences.

To construct the training pipeline, we populate a transition database with historical episodes. For any given target observation-action pair (o_t, a_t), we utilise a dense vector retrieval mechanism to query the database for semantically similar transitions. However, naive retrieval in sequential environments risks severe data contamination. If left unrestricted, a retriever might simply fetch immediately adjacent frames from the same episode, allowing the model to trivially reconstruct the temporal sequence rather than learning underlying environment dynamics.

To prevent this leakage while maximising the utility of the training data, we implement an episode-level masking filter. If a target transition originates from episode $\mathcal{E}_i$, the retriever is constrained to search exclusively within the disjoint set of all available episodes excluding $\mathcal{E}_i$. This strict masking guarantees that the retrieved context acts as a generalised heuristic rather than a direct answer key, preventing intra-episode autocorrelation and ensuring rigorous train-set discipline.

The out-of-episode retrieved transitions are then compiled into a structured prompt alongside the target o_t and a_t. We perform supervised fine-tuning on an LM, optimising the standard autoregressive loss on the generation of the ground-truth next observation o_{t+1}. By training the model explicitly on retrieval-augmented inputs, the model learns to optimally cross-reference its internal parametric knowledge with the non-parametric context provided by the retrieved demonstrations, yielding a robust and generalizable world model.

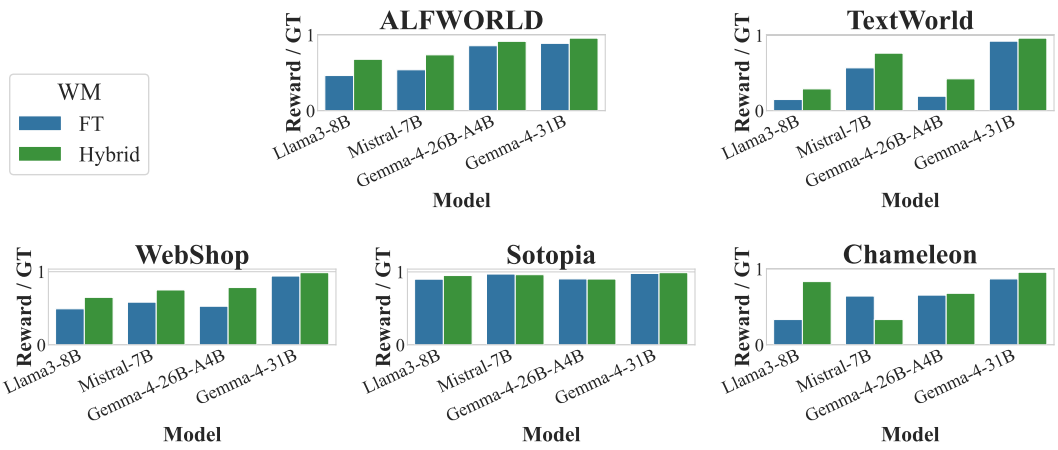

Figure 4: Our hybrid approach combines the benefits of fine-tuning and RAG, outperforming other approaches on 17/20 settings at $p < 0.05$. For the complete results with CIs, see Appendix D.

Results: Our hybrid approach outperforms on 17/20 settings, resolving the bottlenecks of isolated fine-tuning and naive retrieval. The method reaches 77% of the total reward obtainable by the oracle, substantially outperforming the next-best baseline (60%). Furthermore, on 45% of all environment-model combinations, the hybrid model exceeds 90% of oracle performance, showing that fusing internal weight updates with non-parametric context yields highly robust world models.

7 Conclusion

In this work, we perform the first systematic comparison of RAG and fine-tuning for the task of LM-based world modelling. We show that unlike tasks like knowledge acquisition, fine-tuning proves consistently superior to RAG, with only minor exceptions. We demonstrate that while both approaches benefit from increased exploration, RAG is more data-efficient. Our error analysis reveals substantial retriever errors, suggesting that adapting the retriever to the specific environment could yield substantial performance gains. We investigate alternative paradigms of encoding transitions and show that a hierarchical scheme outperforms the classic encoding method. Finally, we introduce a hybrid world model system that combines fine-tuning and RAG to consistently outperform all other baselines.

REFERENCES

Muhammad Arslan, Hussam Ghanem, Saba Munawar, and Christophe Cruz. A survey on rag with llms. *Procedia computer science*, 246:3781–3790, 2024.

Angels Balaguer, Vinamra Benara, Renato Luiz de Freitas Cunha, Todd Hendry, Daniel Holstein, Jennifer Marsman, Nick Mecklenburg, Sara Malvar, Leonardo O Nunes, Rafael Padilha, et al. Rag vs fine-tuning: pipelines, tradeoffs, and a case study on agriculture. *arXiv preprint arXiv:2401.08406*, 2024.

Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Hyungjoo Chae, Namyoung Kim, Kai Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. In *International Conference on Learning Representations*, volume 2025, pp. 63707–63738, 2025.

Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Workshop on computer games*, pp. 41–75. Springer, 2018.

Thomas Palmeira Ferraz, Romain Deffayet, Vassilina Nikoulina, Hervé Déjean, and Stéphane Clinchant. Retrieval-augmented llm agents: Learning to learn from experience. *arXiv preprint arXiv:2603.18272*, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, et al. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3):440, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *ArXiv*, abs/2310.06825, 2023. URL https://api.semanticscholar.org/CorpusID:263830494.

Mustafa O Karabag, Jan Sobotka, and Ufuk Topcu. Do llms strategically reveal, conceal, and infer information? a theoretical and empirical analysis in the chameleon game. *arXiv preprint arXiv:2501.19398*, 2025.

Minsoo Kim and Seung-won Hwang. Coex–co-evolving world-model and exploration. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 21629–21651, 2025.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu (eds.), *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3250/.

Yixia Li, Hongru Wang, Jiahao Qiu, Zhenfei Yin, Dongdong Zhang, Cheng Qian, Zeping Li, Xiaoteng Ma, Guanhua Chen, and Heng Ji. From word to world: Can large language models be implicit text-based world models? In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8084–8111, 2026.

Kai Mei, Jiang Guo, Shuaichen Chang, Mingwen Dong, Dongkyu Lee, Xing Niu, and Jiarong Jiang. R-wom: Retrieval-augmented world model for computer-use agents. *arXiv preprint arXiv:2510.11892*, 2025.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. Fine-tuning or retrieval? comparing knowledge injection in llms. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 237–250, 2024.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 2463–2473, 2019.

Bhagyajit Pingua, Adyakanta Sahoo, Meenakshi Kandpal, Deepak Murmu, Jyotirmayee Rautaray, Rabindra Kumar Barik, and Manob Jyoti Saikia. Medical llms: fine-tuning vs. retrieval-augmented generation. *Bioengineering*, 12(7):687, 2025.

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International conference on machine learning*, pp. 8583–8592. PMLR, 2020.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 1(11):6, 2025.

Heydar Soudani, Evangelos Kanoulas, and Faegheh Hasibi. Fine tuning vs. retrieval augmented generation for less popular knowledge. In *Proceedings of the 2024 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pp. 12–22, 2024.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. *Advances in neural information processing systems*, 28, 2015.

Mengyao Xu, Wenfei Zhou, Yauhen Babakhin, Gabriel Moreira, Ronay Ak, Radek Osmulski, Bo Liu, Even Oldridge, and Benedikt Schifferer. Omni-embed-nemotron: A unified multimodal retrieval model for text, image, audio, and video. *arXiv preprint arXiv:2510.03458*, 2025.

Chang Yang, Xinrun Wang, Qinggang Zhang, Qi Jiang, and Xiao Huang. Efficient integration of external knowledge to llm-based world models via retrieval-augmented generation and reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 9484–9501, 2025.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Xuan Zhang, Wenxuan Zhang, See-Kiong Ng, and Yang Deng. Self-evolving world models for llm agent planning. *arXiv preprint arXiv:2606.30639*, 2026.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. Sotopia: Interactive evaluation for social intelligence in language agents. In *International Conference on Learning Representations*, volume 2024, pp. 40975–41019, 2024.

A Environment Implementation Details

Across all five environments, the world model (WM) is trained to predict the agent's next observation given the interaction history and a candidate action; unless otherwise noted below, the training target is simply the environment's true next observation.

A.1 Embodied Navigation: Observation Shaping

ALFWorld. Rather than exposing the raw simulator observation on its own, each observation given to the agent is augmented with the current room description, the current inventory, and the full list of admissible commands, so that the agent always has a complete, self-contained view of the state at every step. On reset, the environment's introductory banner text is discarded and the task instruction is extracted and surfaced separately from the observation, so that the very first observation of an episode has the same shape as every subsequent one. No truncation is applied to any of these fields.

TextWorld. The same augmentation strategy is used (room description, inventory, admissible commands appended to the observation), with one deliberate deviation: the *look* command is removed from the admissible-command list before it is shown to the agent, since taking that action would simply reproduce the room description already present in the observation and offers no new information. As with ALFWorld, introductory/banner text is dropped at reset and replaced with the task objective, so reset and step observations share a consistent format.

WebShop. Observations are shaped in three ways before being shown to the agent. First, the shopping instruction — which the underlying environment otherwise repeats verbatim inside every observation — is removed, since it is instead surfaced once as a fixed task description for the episode. Second, on the terminal observation, the trailing reward-summary text that the environment appends is stripped, since reward is already tracked numerically by the wrapper rather than needing to be parsed out of the observation text. Third, the set of currently available actions (whether a search is possible, and which clickable page elements are currently valid) is appended explicitly to the observation as an action menu, so the agent's action space at each step is stated rather than implicit. No additional truncation or markup handling is applied; the underlying text rendering of search-results and product pages is otherwise used as-is.

A.2 Adversarial Negotiation: Sotopia

Setup. Each episode is a two-party conversation grounded in a scenario and a pair of character profiles (physical/social context, occupation, personality, private goal, and a piece of secret information each party is not expected to reveal unless it serves them), drawn from the public Sotopia scenario dataset. The agent under evaluation always plays one fixed conversational role; the other party is a separate, non-evaluated language model instantiated specifically to play the counterpart for that episode.

Prompting. Each turn, the speaking party is prompted with the scenario description, its own persona (name, age, occupation, publicly known information, personality and values), its private goal, its secret information, and the conversation so far, and is asked to produce the single best next utterance to advance its own goal, or to end the interaction if it has reached a natural conclusion.

Action and observation space. The action space is an unconstrained natural-language utterance, or an explicit "end the conversation" action available to either party. After each of the agent's turns, the observation returned is the counterpart's reply since the agent's last turn; the agent's own utterance is not echoed back, since it is already recorded as the action that produced that step.

Termination and turn budget. An episode ends when either party chooses to end the conversation, or after a fixed budget of ten player turns has elapsed — chosen to allow conversations to develop fully before being cut off.

Reward. At the end of an episode, a separate judging language model rates the evaluated party's goal achievement on a 0–10 scale, conditioned on the scenario, that party's persona and goal, and the full conversation transcript; this score is rescaled to $[0, 1]$ to form the scalar reward.

Data splits. Scenarios are partitioned by their labeled interaction type. Negotiation scenarios are split 80/20 by scenario into train and test sets under a fixed random seed. Scenarios labeled as competitive (rather than negotiation) are held out in their entirety and used as a distribution-shift evaluation split, disjoint from both the negotiation train and test scenarios.

World-model prediction task. Given the conversation history, the current observation, and the agent's proposed next utterance as the candidate action, the WM is trained to predict the counterpart's resulting reply — i.e. the ground-truth prediction target is literally the counterpart's actual next line of dialogue (or the episode's terminal state, if the proposed utterance ends the conversation).

A.3 Deceptive Communication: Chameleon

Default configuration. The Chameleon environment is general enough to support an arbitrary number of players and an arbitrary number of chameleons among them, but every experiment reported in this paper uses a single fixed configuration: three players and exactly one chameleon. The evaluated agent always occupies the first speaking position and is always a non-chameleon; the chameleon role is always assigned to the second speaking position, leaving the third position as the other (non-chameleon, language-model-driven) player. Three players is in any case the smallest group size for which the game is well defined, and with three players exactly one chameleon is the only valid configuration, so this default is also the game's minimal, canonical form.

Game flow. Each round is drawn from one of several topics (e.g. movies, athletes, companies, countries, superheroes, scientists, food), each with a fixed pool of sixteen candidate secret words per data split. Players speak once each, in a fixed order. Non-chameleon players are shown the secret word and asked to give a single related word that signals their knowledge of it to other non-chameleons without making it obvious to the chameleon; the chameleon does not know the secret word and must produce a plausible-sounding clue by inference from the other clues and the topic alone. Since the evaluated agent always speaks first, it must commit to its clue with no information beyond the topic and secret word, and the round always concludes after everyone has spoken once.

Judge. After every player has given a clue, a separate judging language model is shown the secret word, the full candidate word list for the topic, and all clues in speaking order, and must name exactly one player as the chameleon.

Reward. From the evaluated player's perspective (always a non-chameleon under the default configuration): the reward is $+1$ if the judge correctly names the true chameleon, 0 if the judge instead accuses the other non-chameleon player, and this outcome would additionally be penalized by -1 if the judge accused the evaluated player itself — a case that cannot arise under the fixed default configuration, since the evaluated player's position is never assigned the chameleon role.

World-model prediction task. Because the round always ends after a single exchange of clues, the world model's prediction task is correspondingly single-step: conditioned on the topic, secret word, and the evaluated agent's chosen clue (the candidate action), the WM must predict both the clues subsequently given by the other two players and the judge's eventual verdict. Rather than using the literal next observation as the training target, the ground-truth target is constructed specifically to isolate these two pieces of information: the other players' clues, and a short natural-language statement of whether the judge's guess correctly caught the true chameleon or was misled into accusing the wrong player.

B World Model Deployment and Evaluation

At deployment time, the policy-agent LM is prompted with the environment description, task description, interaction history, and current observation, and asked to produce a ranked list of m can-

didate actions in a single call, ordered from most to least preferred, rather than a single committed action. The base framing shared by every prompt to the policy-agent LM is:

```
You are an agent in [ENVIRONMENT DESCRIPTION]
Your task is:  [TASK_DESCRIPTION]
The history of your interactions with the environment is:
[HISTORY]
Your current observation of the environment is:
[OBS]
```

and the specific instruction appended to it to elicit the ranked candidate list is:

```
Given this, what are the best actions to take next?  Give
a ranked list of the top [K] action candidates, in order of
preference, with the most preferred action first.  Give your
output in the following format:
Reasoning:  [your reasoning here, which will not be included
in the final output]
Action 1:  [the most preferred action, exactly as it should
be input into the environment, and nothing else]
Action 2:  [the second most preferred action, exactly as it
should be input into the environment, and nothing else]
...
Action K: [the K-th most preferred action, exactly as it
should be input into the environment, and nothing else]
[STOP]
```

Each of the m candidates is then passed to the world model under evaluation, which independently predicts the resulting next observation for that candidate given the same history and current observation. The prompt used to elicit each prediction is:

```
You are a world model in [ENVIRONMENT DESCRIPTION]
[RETRIEVED TRANSITION STRING]
The history of the agent's interactions with the environment
is:
[HISTORY]
The current observation of the environment is:
[OBS]
The agent is considering taking this action:
[ACTION]
Given all of this what is the most likely change in state or
next observation that the agent will see?
First reason step by step about the mechanics, then describe
the resulting state change.
Give your answer in the following format:
Reasoning:  [your reasoning here, which will not be included
in the final output]
Prediction:  [a concise description of what changed and what
the new state of the environment looks like]
[STOP]
```

(the retrieved-transition block is populated only for retrieval-augmented world models, and is left empty otherwise.) This yields m predicted observations, one per candidate action.

Each predicted observation is then scored by the same policy-agent LM in a separate call, using the same base framing above, followed by:

```
You then took action [ACTION] and a state change occurred in
the environment, giving you the next observation:
[STATE_CHANGE]
Given this, how beneficial is this state change for achieving
your task?  Rate on a scale of 1 to 10, where 1 means active
regression from the goal, 3 means no real progress, 5 means
some progress, 7 means significant progress, and 10 means
excellent progress or goal achieved.
First reason step by step about how this state change relates
to your task, then give your final score.
Answer in the following format:
Reasoning:  <think about how this state change relates to
the task goal>
Score:  <integer 1 to 10>
[STOP]
```

The candidate action whose predicted observation receives the highest score is selected and executed in the real environment. If the world model fails to produce a usable prediction for any of the m candidates, the agent falls back to executing its own top-ranked candidate directly, without invoking the scoring step.

In practice we set $m = 5$. The policy-agent LM is held frozen throughout — it is never fine-tuned or otherwise updated over the course of these experiments — and is used identically for candidate-action proposal and for scoring predicted observations. We evaluate this pipeline with four different policy-agent LMs: Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Mistral-7B-Instruct (Jiang et al., 2023), Gemma-4-26B-A4B-Instruct, and Gemma-4-31B-Instruct (Team et al., 2025).

C EXPLORATION

Training buffers for the world models are constructed by rolling out an agent in each training environment. At each step, with probability ϵ the agent's action is instead generated by an exploratory variant of the same policy-agent LM rather than its default goal-directed behavior; with probability $1 - \epsilon$ it behaves greedily as usual. In both cases the action is still produced by prompting the LM with the same base framing and output format shown in Appendix B (single action or ranked candidate list, depending on whether a world model is active that step) — exploration is implemented entirely by substituting the task instruction given to the model, not by sampling actions uniformly at random.

For most environments, the exploratory variant entirely replaces `[TASK_DESCRIPTION]` in the base framing above with a single generic exploration instruction:

```
Take interesting exploratory actions to probe various aspects
of the environment.  Avoid doing the same thing repeatedly,
and try to think creatively and be very wide in what you
consider doing (but within any specified list of allowed
actions).
```

Two environments instead use a specialized exploration instruction tailored to their setting, appended to (rather than replacing) the normal task instruction, in the form "`[normal task instruction]` [line break] For this turn, try an exploratory approach: `[specialized instruction]` ". In Chameleon, the specialized instruction is:

```
Be radically creative with your clue.  Ignore what seems
safe or obvious | pick something that no one would expect,
that comes from a completely different angle, that might even
seem wrong at first glance but is secretly defensible.  The
```

```
goal is maximum surprise while still being technically valid.
Push hard against your first instinct.
```

and in Sotopia, it is:

```
Throw out the safe playbook entirely.  Take a bold,
unexpected gamble | try something you normally wouldn't dare,
a completely different angle, an unconventional tactic that
could pay off big or fail spectacularly.  The worst outcome
is being boring.  Commit fully to the unusual choice.
```

This exploration mechanism is applied uniformly across all environments and is independent of which world model, if any, is being used to select actions during rollout. It is used only when constructing training buffers; at evaluation time ϵ is fixed to zero, so that reported results always reflect purely greedy behavior. We sweep $\epsilon \in \{0, 0.25, 0.5, 0.75, 1.0\}$ when constructing training buffers, in order to study how the mixture of exploratory and greedy trajectories in the training buffer affects the resulting world model's quality.

D Complete Results

We show complete results, including 95% confidence intervals from a bootstrap t-test below (Table 2). We also show the fidelity measurements for the Llama3-8B model on all explored metrics (Table 3). Limited variance for BLEU and BertScore suggest that they are poor metrics for WM fidelity.

E Hardware

Experiments were run on a single cluster with 8xA100 NVIDIA GPUs.

F Use of LLMs

We used LLMs for the following:

- Help with code implementation (Claude Code). No design decisions were made unilaterally by the LLM.
- Appendix writing: Claude Code was used to verbalize the experimental details and hyperparameters during the writing of certain Appendix sections. All text was checked by the authors to confirm veracity.
- Error and spell checking.

We did not use LLMs for any of the following:

- Ideation/project design
- Interpretation of results

Environment	Model	No	ZS		FS		FT		RAG		Hybrid	
		Util	Util	Fid	Util	Fid	Util	Fid	Util	Fid	Util	Fid
ALF	Llama	0.36 ± 0.02	0.43 ± 0.01	4.15 ± 0.02	0.25 ± 0.06	3.96 ± 0.08	$\underline{0.46} \pm 0.01$	$\underline{4.78} \pm 0.19$	0.36 ± 0.00	4.09 ± 0.02	$\mathbf{0.68} \pm 0.06$	$\mathbf{7.42} \pm 0.79$
	Mistral	0.00 ± 0.12	0.30 ± 0.03	3.95 ± 0.05	0.20 ± 0.06	3.82 ± 0.08	$\underline{0.54} \pm 0.05$	$\underline{4.65} \pm 0.16$	0.39 ± 0.03	4.12 ± 0.05	$\mathbf{0.74} \pm 0.06$	$\mathbf{7.55} \pm 0.87$
	Gem-26B	0.79 ± 0.01	0.80 ± 0.00	4.33 ± 0.16	0.76 ± 0.01	4.16 ± 0.10	$\underline{0.86} \pm 0.01$	$\underline{5.10} \pm 0.07$	0.81 ± 0.00	4.87 ± 0.07	$\mathbf{0.92} \pm 0.02$	$\mathbf{6.12} \pm 0.31$
	Gem-31B	0.82 ± 0.01	0.67 ± 0.03	4.23 ± 0.47	0.72 ± 0.06	5.01 ± 0.03	$\underline{0.89} \pm 0.01$	$\underline{6.18} \pm 0.32$	0.86 ± 0.01	5.11 ± 0.03	$\mathbf{0.96} \pm 0.02$	$\mathbf{7.84} \pm 0.50$
Text	Llama	0.00 ± 0.04	$\underline{0.16} \pm 0.00$	3.40 ± 0.19	0.06 ± 0.03	2.87 ± 0.32	0.15 ± 0.00	4.04 ± 0.03	0.11 ± 0.01	$\underline{4.15} \pm 0.03$	$\mathbf{0.29} \pm 0.04$	$\mathbf{4.42} \pm 0.08$
	Mistral	0.494 ± 0.006	$\underline{0.634} \pm 0.020$	3.45 ± 0.19	0.504 ± 0.028	2.90 ± 0.33	0.569 ± 0.005	4.10 ± 0.03	0.552 ± 0.005	$\underline{4.20} \pm 0.03$	$\mathbf{0.764} \pm 0.039$	$\mathbf{4.55} \pm 0.10$
	Gem-26B	0.000 ± 0.087	$\underline{0.219} \pm 0.008$	3.61 ± 0.20	0.144 ± 0.014	3.13 ± 0.29	0.193 ± 0.007	4.28 ± 0.02	0.168 ± 0.007	$\underline{4.36} \pm 0.02$	$\mathbf{0.424} \pm 0.061$	$\mathbf{5.17} \pm 0.24$
	Gem-31B	0.848 ± 0.017	0.755 ± 0.055	5.20 ± 0.13	0.631 ± 0.075	4.82 ± 0.23	$\underline{0.924} \pm 0.006$	$\underline{5.86} \pm 0.07$	0.904 ± 0.006	5.63 ± 0.07	$\mathbf{0.964} \pm 0.012$	$\mathbf{6.84} \pm 0.29$
Web	Llama	0.04 ± 0.02	0.07 ± 0.01	4.77 ± 0.09	0.09 ± 0.05	3.78 ± 0.59	$\underline{0.49} \pm 0.05$	$\underline{6.06} \pm 0.23$	0.24 ± 0.05	5.08 ± 0.09	$\mathbf{0.65} \pm 0.05$	$\mathbf{6.84} \pm 0.23$
	Mistral	0.00 ± 0.10	0.17 ± 0.05	4.65 ± 0.15	0.25 ± 0.02	3.85 ± 0.48	$\underline{0.58} \pm 0.05$	$\underline{6.15} \pm 0.24$	0.33 ± 0.02	5.15 ± 0.15	$\mathbf{0.75} \pm 0.05$	$\mathbf{6.95} \pm 0.24$
	Gem-26B	0.08 ± 0.02	0.12 ± 0.03	4.81 ± 0.16	0.17 ± 0.05	4.11 ± 0.42	$\underline{0.53} \pm 0.05$	$\underline{6.27} \pm 0.19$	0.35 ± 0.05	5.34 ± 0.16	$\mathbf{0.78} \pm 0.08$	$\mathbf{6.89} \pm 0.19$
	Gem-31B	0.904 ± 0.006	0.601 ± 0.052	4.01 ± 0.17	0.687 ± 0.130	4.29 ± 0.46	$\underline{0.940} \pm 0.005$	$\underline{6.43} \pm 0.05$	0.923 ± 0.005	5.81 ± 0.19	$\mathbf{0.987} \pm 0.014$	$\mathbf{6.58} \pm 0.05$
Cha	Llama	0.17 ± 0.05	-0.50 ± 0.24	2.56 ± 0.08	-0.10 ± 0.08	2.70 ± 0.25	-0.52 ± 0.01	3.54 ± 0.10	$\underline{0.33} \pm 0.05$	$\underline{3.87} \pm 0.10$	$\mathbf{0.83} \pm 0.15$	$\mathbf{4.88} \pm 0.30$
	Mistral	-4.50 ± 0.37	-3.74 ± 0.20	2.45 ± 0.09	-3.07 ± 0.20	2.60 ± 0.26	-3.88 ± 0.09	3.45 ± 0.15	$\mathbf{0.64} \pm 0.09$	$\underline{3.95} \pm 0.05$	$\underline{0.33} \pm 0.09$	$\mathbf{4.13} \pm 0.05$
	Gem-26B	0.501 ± 0.006	0.070 ± 0.246	3.22 ± 0.12	0.480 ± 0.006	3.42 ± 0.08	0.480 ± 0.000	3.70 ± 0.08	$\underline{0.655} \pm 0.007$	$\underline{4.31} \pm 0.11$	$\mathbf{0.679} \pm 0.007$	$\mathbf{4.66} \pm 0.11$
	Gem-31B	0.78 ± 0.01	-0.05 ± 0.27	3.28 ± 0.33	0.41 ± 0.20	4.16 ± 0.10	0.73 ± 0.01	3.83 ± 0.10	$\underline{0.87} \pm 0.03$	$\underline{5.17} \pm 0.30$	$\mathbf{0.96} \pm 0.03$	$\mathbf{6.24} \pm 0.32$
Soto	Llama	0.75 ± 0.04	0.89 ± 0.00	4.50 ± 0.03	0.82 ± 0.02	4.19 ± 0.19	$\underline{0.90} \pm 0.00$	4.61 ± 0.03	0.86 ± 0.01	$\underline{4.77} \pm 0.04$	$\mathbf{0.95} \pm 0.01$	$\mathbf{4.90} \pm 0.04$
	Mistral	0.950 ± 0.004	0.963 ± 0.000	4.40 ± 0.05	0.956 ± 0.004	4.24 ± 0.10	$\mathbf{0.970} \pm 0.000$	4.58 ± 0.05	$\underline{0.969} \pm 0.000$	$\underline{4.86} \pm 0.02$	0.961 ± 0.000	$\mathbf{4.92} \pm 0.02$
	Gem-26B	0.850 ± 0.010	0.868 ± 0.001	4.61 ± 0.09	0.869 ± 0.006	4.42 ± 0.11	$\mathbf{0.903} \pm 0.000$	4.92 ± 0.05	0.887 ± 0.004	$\mathbf{5.12} \pm 0.01$	$\underline{0.902} \pm 0.000$	$\underline{5.07} \pm 0.01$
	Gem-31B	0.96 ± 0.00	0.96 ± 0.00	4.73 ± 0.02	0.64 ± 0.19	4.62 ± 0.07	$\underline{0.98} \pm 0.00$	4.81 ± 0.02	0.97 ± 0.00	$\underline{4.87} \pm 0.01$	$\mathbf{0.99} \pm 0.00$	$\mathbf{4.92} \pm 0.01$

Table 2: Utility and Fidelity of LM-based WMs. **Bold** marks the best value in each row per metric; underline marks the second best. Fine-tuning is the superior standalone method, achieving superior utility on 15/20 settings. RAG maintains a consistent advantage over the baseline approaches and outperforms fine-tuning on the Chameleon environment. Hybrid outperforms all other methods consistently, with the only exceptions being a single setting of Chameleon and two Sotopia settings.

Environment	WM	BLEU	BertScore	LM-as-Judge
ALFWORLD	None			-
	ZS	0.00	0.83	4.15
	FS	0.00	0.79	3.96
	FT	0.00	0.80	4.78
	RAG	0.00	0.89	4.09
	Hybrid	0.02	0.85	7.42
	GT			10.00
Chameleon	None	0.00		-
	ZS	0.00	0.81	2.56
	FS	0.02	0.83	2.70
	FT	0.04	0.83	3.54
	RAG	0.02	0.85	3.87
	Hybrid	0.03	0.84	4.88
	GT			10.00
Sotopia	None			-
	ZS	0.03	0.84	4.50
	FS	0.03	0.85	4.19
	FT	0.03	0.85	4.61
	RAG	0.18	0.91	4.77
	Hybrid	0.21	0.90	4.90
	GT			10.00
TextWorld	None			-
	ZS	0.01	0.76	3.40
	FS	0.01	0.76	2.87
	FT	0.01	0.76	4.04
	RAG	0.00	0.78	4.15
	Hybrid	0.04	0.77	4.42
	GT			10.00
WebShop	None			-
	ZS	0.01	0.78	4.77
	FS	0.00	0.78	3.78
	FT	0.01	0.77	6.06
	RAG	0.01	0.78	5.08
	Hybrid	0.30	0.90	6.84
	GT			10.00

Table 3: Complete results with all fidelity metrics. Low variance with BLEU and BertScore suggest they are poor metrics for WM-fidelity.